# Optimizing Multi-Relational Factorization Models for Multiple Target Relations

Lucas Drumond
Information Systems and Machine Learning Lab
University of Hildesheim, Germany
ldrumond@ismll.de

Ernesto Diaz-Aviles
IBM Research
Dublin, Ireland
e.diaz-aviles@ie.ibm.com

Lars Schmidt-Thieme
Information Systems and Machine Learning Lab
University of Hildesheim, Germany
schmidt-thieme@ismll.de

Wolfgang Nejdl
L3S Research Center
University of Hannover, Germany
nejdl@L3S.de

## ABSTRACT

Multi-matrix factorization models provide a scalable and effective approach for multi-relational learning tasks such as link prediction, Linked Open Data (LOD) mining, recommender systems and social network analysis. Such models are learned by optimizing the sum of the losses on all relations in the data. Early models address the problem where there is only one target relation for which predictions should be made. More recent models address the multi-target variant of the problem and use the same set of parameters to make predictions for all target relations. In this paper, we argue that a model optimized for each target relation individually has better predictive performance than models optimized for a compromise on the performance on all target relations. We introduce specific parameters for each target but, instead of learning them independently from each other, we couple them through a set of shared auxiliary parameters, which has a regularizing effect on the target specific ones. Experiments on large Web datasets derived from DBpedia, Wikipedia and BlogCatalog show the performance improvement obtained by using target specific parameters and that our approach outperforms competitive state-of-the-art methods while being able to scale gracefully to big data.

**Categories and Subject Descriptors:** H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval–Information filtering; I.2.6 [Artificial Intelligence]: Learning–Parameter learning

**General Terms:** Algorithms; Experimentation; Measurement.

**Keywords:** Statistical inference; Relational learning; Factorization Models.


*CIKM'14,* November 3–7, 2014, Shanghai, China.

http://dx.doi.org/10.1145/2661829.2662052.

## 1. INTRODUCTION

A lot of work has been devoted to analyzing and learning from data in a table format where instances are represented as a feature vector and have a label associated with them. Although such approaches have been quite sucessful, new models able to cope with richer structures in the data are needed. Most data available on the Web have a complex graph structure comprising different relations (e.g., edge types). Thus, mining multi-relational data with noise, partial inconsistencies, ambiguities, or duplicate entities, has gained relevance in the last years and found applications in a number of tasks such as link prediction [14], Resource Description Framework (RDF) mining [6], entity linking [18], recommender systems [10], and natural language processing [9]. However, new paradigms are still needed for statistical and computational inference based on multi-relational data.

Recently, multi-relational factorization models have shown to scale well while providing good predictive performance and are currently considered the state-of-the-art for Statistical Relational Learning (SRL) tasks [13, 19]. Factorization models for multi-relational data associate entities and relations with latent feature vectors and define predictions about new relationships through operations on these vectors (e.g., dot products). A number of factorization models define one single relation for which predictions should be made, called the *target* relation, while the other relations are used as side information (*auxiliary* relations) [10, 12, 19, 22]. Consider for instance the scenario of online social networks (OSNs), such as Facebook, YouTube, or Flickr, which encourage users to create connections between themselves or to interesting items (e.g., songs, videos, or news items). The social information (connection between users) can be exploited by recommender systems to provide better recommendations of items of interest (connections between users and items) [10].

In order to illustrate how multi-relational factorization models work, we introduce a running example used across the paper. Consider a social media website where users can follow other users (much like in Twitter), be friends with

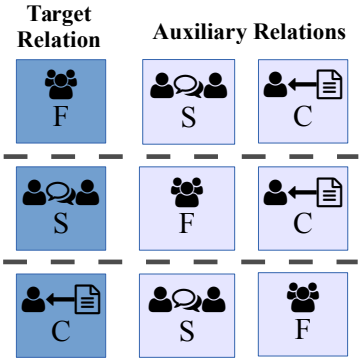


Figure 1: **In this Multi-Relation and Multi-Target example there are three relations: *follows* F, *social* S and *consumes* C, between two entities *users* (👤) and *news items* (🗎). This example shows the corresponding three cases for which each relation is acting as a target and the rest ones as auxiliary.**

other users (forming a social graph) and consume products, e.g., read news items. In this example there are two entity types, namely users $U$ and news items $N$, and three relations: (i) *follows* $F := U \times U$, (ii) the *social* relationship $S := U \times U$ and (iii) the product *consumption* (e.g. reading of news items) $C := U \times N$, as depicted in Figure 1.

Historically, the first multi-relational learning factorization models were concerned with making predictions for a single *target* relation based on information provided by a set of other relations which we refer to as *auxiliary*. However, in domains with many potential target relations, a more interesting model class is required in order to make predictions for *all* targets, e.g., in the context of recommender systems, one is not only interested in recommending news items to a user but also recommending other users whom she might want to follow, or be friends with.

Another example of a task where this is important is the mining of Linked Open Data bases like DBPedia, for instance supporting probabilistic queries on such databases and providing estimates of facts that are neither explicitly stated in the knowledge base nor can be inferred from logical entailment [6, 14]. Optimizing the predictions for a number of relations can be seen as a prediction task with multiple target variables. State-of-the-art factorization models approach the problem by sharing the parameters used for predicting all target relations. Instances of such approaches are RESCAL [13, 14], MOF-SRP [9] and SME [3], which share entity specific parameters among all relations in the data. This way, the best solution for the optimization problem is a compromise of the performance on all relations. Although most of these models have been evaluated on multi-target settings, none of them have explicitly investigated the problem of how to optimize each target relation *individually* instead of learning the optimal performance compromise on all relations.

When optimizing a model for one specific relation (so called target relation), neglecting the information on the other relations leads to suboptimal results. Thus state-of-the-art models downweight the contribution of the auxiliary relations to the overall loss function. Thus, the model exploits all the relational information available but it is still optimized for the target. When optimizing for multiple target relations, we propose to learn a set of single target models each one optimized for one relation while downweighting the other ones. We call this approach *Decoupled Target Specific Features Multi-Target Factorization (DMF)*.

One drawback of this approach is that the number of parameters to be learned grows too fast with the number of relations. However, when learning a model with DMF, a number of parameters are used only for auxiliary relations and never for predicting the targets. By sharing such parameters among the models for different targets, one can reduce significantly the amount of memory required by the model. This second approach we call *Coupled Auxiliary and Target Specific Features Multi-Target Factorization* (CATSMF). This is the first work to specifically investigate how to optimize multi-relational factorization models for each target relation individually. In summary, the main contributions of this work are:

1. We propose a new factorization approach that optimizes directly for multiple target relations. The novelty of our approach lies in the fact that for the same entities we use different parameters when making predictions for different target relations, thus allowing the model to be optimized for each target relation specifically;

2. We also show that coupling the models for different target relations, by introducing shared parameters for reconstructing relations when they play an auxiliary role, leads to a more memory efficient method and even to better predictive accuracy;

3. We empirically show the advantage of having specific predictive parameters on different relations to the overall loss. Our experiments on real world Web datasets from DBpedia, Wikipedia and BlogCatalog demonstrate that CATSMF outperforms state-of-the-art factorization models and has lower runtime. Furthermore they also provide empirical evidence that taking into account entity type information can have a positive impact on predictive performance.

## 2. PROBLEM FORMULATION

Relational data comprise a set of $R \in \mathbb{N}$ relations among a set of entities $\mathcal{E}$. The data for a given relation $r \in \{1, \ldots, R\}$ can be described as $D_r := \{(e_r, y_r) | e_r \in E_r \wedge y_r \in \mathbb{R}\}$ where $E_r \subseteq \mathcal{E}^{n_r}$ is called the *extension* of relation $r$, $n_r$ denotes its arity, and $y_r$ is a value associated with each observation. In this paper we assume all the relations to be binary, i.e. $n_r = 2$.

Let $\mathcal{X}_r := E_r, \mathcal{Y}_r := \mathbb{R}$ be sets called predictor and target spaces of relation $r$, respectively, for $r = 1, \ldots, R$. The training data for a relation $r$ can be written as $D_r^{\text{train}} \subseteq \mathcal{X}_r \times \mathcal{Y}_r$. Many times $y_r$ denotes the truth value of a given observation and can be encoded as $y_r \in \{0, 1\}$. As an example, imagine a binary relation relating countries to their capitals. Possible observations could be $(\text{Germany}, \text{Berlin}, 1)$ and $(\text{Germany}, \text{Hamburg}, 0)$.

Let $Y_r = \{\hat{y}_r : \mathcal{X}_r \to \mathcal{Y}_r\}$ be the space of all possible prediction models considered and $L_r : \mathcal{P}(\mathcal{X}_r \times \mathcal{Y}_r) \times Y_r \to \mathbb{R}_0^+$ be a loss function, where $\mathcal{P}$ denotes the power set. Given the training data, the multi-relational multi-target prediction problem is to find $R$ models $\hat{y}_r : \mathcal{X}_r \to \mathcal{Y}_r$ s.t. for some test data $D_r^{\text{test}} \subseteq \mathcal{X}_r \times \mathcal{Y}_r$ $(r = 1, \ldots, R)$ stemming from the

same data generating process as the training data and not being used for learning the models $\hat{y}_r$, the test error

$$\text{error}((D_r^{\text{test}})_{r=1,\dots,R}, (\hat{y}_r)_{r=1,\dots,R}) := \frac{1}{R}\sum_{r=1}^{R} L_r(D_r^{\text{test}}, \hat{y}_r)$$

is minimal.

For regression and classification problems, losses $L_r$ usually are defined as a sum of pointwise losses $\ell_r : \mathcal{Y}_r \times \mathcal{Y}_r \to \mathbb{R}_0^+$:

$$L_r(D_r^{\text{test}}, \hat{y}_r) := \frac{1}{|D_r^{\text{test}}|}\sum_{(x,y)\in D_r^{\text{test}}} \ell_r(y, \hat{y}_r(x)).$$

A number of multi-relational datasets consist of positive instances only, e.g., the tuples of entities $\mathcal{E}$ in a subset of the extension of the relation. This means that we only observe a subset of the tuples of the type $(x, y)$ where $y = 1$. In this case we are interested in solving a *ranking* task where prediction functions $Y_r = \{\hat{y}_r : \mathcal{X}_r \to \mathbb{R}\}$ deliver ranking scores and the losses $L_r$ usually are defined pairwise:

$$L_r(D_r^{\text{test}}, \hat{y}_r) := \frac{1}{|D_r^{\text{test}}||\mathcal{X}_r \times \{1\} \setminus (D_r^{\text{train}} \cup D_r^{\text{test}})|}\sum_{(x,1)\in D_r^{\text{test}}} \sum_{(x',1)\in \mathcal{X}_r\times\{1\}\setminus(D_r^{\text{train}}\cup D_r^{\text{test}})} \ell_r(\hat{y}_r(x), \hat{y}_r(x')) \tag{1}$$

with pair ranking score losses $\ell_r : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_0^+$. Finally, problems with additional entity type information can be modeled by choosing $\mathcal{X}_r := \mathcal{E}_r^{(1)} \times \mathcal{E}_r^{(2)}$ with $\mathcal{E}_r^{(1)} \subseteq \mathcal{E}$ and $\mathcal{E}_r^{(2)} \subseteq \mathcal{E}$ being subsets of entities that can possibly be related through relation $r$ as subjects and objects respectively.

# 3. RELATED WORK ON MULTI-TARGET FACTORIZATION

In this section we discuss related works on Multi-Target Factorization. We introduce the state-of-the-art methods in this field and position our model.

Early work on Statistical Relational Learning (SRL) aims at statistically modeling relational data [7]. SRL combines graphical models such as Bayesian and Markov networks, with knowledge representation formalisms such as first order logic for an accurate modeling of the relationships [17]. Another approach to SRL is multi-relational factorization models, which embed entities into a latent space and reconstruct the relations through operations on this space. These embeddings are shared across the relations.

In order to discuss existing work, we make use of the running example of a social media website introduced in Section 1, where users can follow other users (much like in Twitter), be friends with other users (forming a social graph) and consume products, e.g., read news items.

Early factorization approaches for multi-relational learning were concerned with making predictions for a single *target* relation based on information provided by a set of other relations which we refer to as *auxiliary*. These approaches learn the model parameters by optimizing the sum over the losses on each relation. In this way, the minimization of the loss on the auxiliary relations acts as a regularization term for the parameters. The overall loss is then a weighted sum of losses and the parameters are learned by optimizing the following loss function:

$$f(\Theta) := \sum_{r=1}^{R} \alpha_r L_r(D_r, \hat{y}_r(D_r; \Theta)) + \lambda||\Theta||_2^2 \tag{2}$$

where $\alpha_r \in \mathbb{R}_0^+$ is a hyperparameter for the contribution of the reconstruction of $r$ to the overall loss and $\lambda$ is the regularization constant for the model parameters $\Theta$. As already observed in previous work [10, 19], if there is a single target relation $t$, better results are achieved by having a weighted sum over the losses. The idea behind this is that different relations contain useful information about the others. For instance knowing which other users a given user follows might give some indication of which kind of news she is interested in. Each weight $\alpha_r$ models how much each relation contributes for the prediction of the target one.

However, in real world scenarios one is often interested in a model that is optimized for making predictions for *all* the relations. In our social media scenario one is not only interested in recommending news items to a user but also recommending other users whom she might want to follow, or be friends with. The differences between various multi-target approaches lie in (i) their parametrization, (ii) the prediction function $\hat{y}$ and (iii) the loss function $L_r$ for which each relation is optimized. Singh and Gordon provide a unified view of such approaches in a model class they call Collective Matrix Factorization (CMF) [19]. CMF uses the following prediction function:

$$\hat{y}_r(x_1, x_2) := \varphi(x_1)^\top \varphi(x_2) \tag{3}$$

where $\varphi : \mathcal{E} \to \mathbb{R}^k$ associates latent features with every entity $x \in \mathcal{E}$, with $k \in \mathbb{N}$ being the number of latent features. Approaches like the Coupled Matrix and Tensor Factorization (CMTF) [1] and MetaFac [11] extended such models to deal with higher arity relations (i.e. relations between more than two entities). For the purposes of this work, we restrict ourselves to relations of arity two, in which case the prediction model of CMTF reduces to the same as in Equation 3. MetaFac on its turn introduces a set of global features which for the arity two case can be described as $\Phi \in R^{k\times k}$, a diagonal matrix which is the same across the predictions for all the relations. The MetaFac prediction function for relations of arity two can be given as:

$$\hat{y}_r(x_1, x_2) := \varphi(x_1)^\top \Phi \varphi(x_2) \tag{4}$$

Such models have the advantage of computational ease but can poorly handle relations with a signature clash, i.e., different relations between the same entity types like the *friends* and *follows* relation from our example. For instance such a model would predict that every user who follows *Barack Obama* is also a friend of his.

One way to cope with this issue is to associate feature matrices $\Phi_r \in \mathbb{R}^{k\times k}$ with each relation:

$$\hat{y}_r(x_1, x_2) := \varphi(x_1)^\top \Phi_r \varphi(x_2) \tag{5}$$

If the relation features $\Phi_r$ are diagonal matrices, this model is equivalent to a PARAFAC tensor decomposition [8]. The Semantic Matching Energy (SME) model [3] also uses this approach although with a slightly different prediction function. This solves the signature clash issue but another lim-

itation remains. One can easily see that, for the models from Equation 3, Equation 4 and Equation 5 (with diagonal $\Phi$ matrices), $\hat{y}_r(x_1, x_2) = \hat{y}_r(x_2, x_1)$. This is an issue when dealing with asymmetric relations, i.e., relations where $y(x_1, x_2) \neq y(x_2, x_1)$ like the *follows* relation in our example.

In this case the model would predict that *Shakira* is interested in following every user that follows her, which is not necessarily true. Using full instead of diagonal matrices for $\Phi_r$ yields a model capable of dealing with this problem. This is the prediction model used by RESCAL [13]. The disadvantage of this model is that it comes at the expense of computational cost, both from processing time and memory standpoints. Another model which we will refer to as Multiple Order Factorization with Shared Relation Parameters (MOF-SRP)[1] [9] aims at reducing the memory requirements by defining relation features as outer products of feature vectors.

None of the aforementioned state-of-the-art approaches make any distinction between target and auxiliary relations, and all of them use the same parameters for predicting all the targets, so that the learned parameters are a compromise for the performance over all targets, but not for each specific one. In our work, however, we propose to combine the idea of shared parameters and learn individual entity embeddings for different target relations which in turn leads to better predictive performance.

# 4. OPTIMIZING MODELS FOR MULTIPLE TARGET RELATIONS

In state-of-the-art methods, the parameters are learned in such a way that they are optimized for the best performance compromise over all relations and not for the best performance on each relation individually (cf. Section 3). To see how this is suboptimal for a general model class, we first present an approach that we call *Decoupled Target Specific Features Multi-Target Factorization* or *DMF* as a stepping stone and introduction to our core contribution in this paper, namely: the *Coupled Auxiliary and Target Specific Features Multi-Target Factorization (CATSMF)* model.

## DMF

Let $\varphi$ be the set of model parameters and $y_r(\cdot;\varphi)$ a prediction model for relation $r$ parametrized with $\varphi$. Also, let the set of parameters with the best prediction performance on relation $r$ be denoted by $\varphi_r^*$. Such parameters are defined as:

$$\varphi_r^* := \arg\min_{\varphi} L_r(D_r, \hat{y}_r(\cdot;\varphi)).$$

Now, suppose the data comprise two distinct target relations, namely $r$ and $s$. State-of-the-art models solve this problem as follows:

$$\varphi^* := \arg\min_{\varphi} \left(L_r(D_r, \hat{y}_r(\cdot;\varphi)) + L_s(D_s, \hat{y}_s(\cdot;\varphi))\right).$$

Now one would expect that $\varphi_r^* \neq \varphi_s^*$. However, by optimizing an objective function like in Equation 2 one is constrained to solutions of the form $\varphi_r = \varphi_s = \varphi^*$.
By definition,

$$L_r(D_r, \hat{y}_r(\cdot;\varphi_r^*)) \leq L_r(D_r, \hat{y}_r(\cdot;\varphi^*))$$

and

$$L_s(D_s, \hat{y}_s(\cdot;\varphi_s^*)) \leq L_s(D_s, \hat{y}_s(\cdot;\varphi^*))$$

from which it follows that

$$L_r(D_r, \hat{y}_r(\cdot;\varphi_r^*)) + L_s(D_s, \hat{y}_s(\cdot;\varphi_s^*)) \leq L_r(D_r, \hat{y}_r(\cdot;\varphi^*)) + L_s(D_s, \hat{y}_s(\cdot;\varphi^*)) \ .$$

This means that using parameters optimized specifically for each target relation is, in the *worst* case, at least as good as having one common set of parameters optimized for all relations. Thus a more appropriate solution is to learn one model for each target relation, an approach that we call *Decoupled Target Specific Features Multi-Target Factorization (DMF)*:

$$\varphi_r^* := \arg\min_{\varphi_r} L_r(D_r, \hat{y}_{r,r}(\cdot;\varphi_r)) + \alpha_{r,s} L_s(D_s, \hat{y}_{r,s}(\cdot;\varphi_r))$$
$$\varphi_s^* := \arg\min_{\varphi_s} L_s(D_s, \hat{y}_{s,s}(\cdot;\varphi_s)) + \alpha_{s,r} L_r(D_r, \hat{y}_{s,r}(\cdot;\varphi_s))$$

with $0 \leq \alpha_{r,s} \leq 1$ and $0 \leq \alpha_{s,r} \leq 1$ and predict using $\hat{y}_{r,r}(\cdot;\varphi_r^*)$ for relation $r$ and $\hat{y}_{s,s}(\cdot;\varphi_s^*)$ for relation $s$.

More generally, let $\hat{y}_{t,r}$ denote the prediction function for a given relation $r$ when another relation $t$ is the target. The loss function of multi-target factorization models can be written as follows:

$$J(\{\varphi_t\}_{t\in 1,\ldots,R}) := \sum_{t=1}^{R}\left(\sum_{r=1}^{R} \alpha_{t,r} L_r(D_r, \hat{y}_{t,r}(D_r;\varphi_t)) + \lambda_t ||\varphi_t||^2\right) \quad (6)$$

Predictions for unseen data points are done using $\hat{y}_t := \hat{y}_{t,t}$ . The functions $\hat{y}_{t,r}$ for $r \neq t$ are called auxiliary reconstructions of relation $r$ for the target relation $t$. $L_r$ is the loss on relation $r$, as defined in Section 2, and $\alpha_{t,r}$ is the importance of relation $r$ when relation $t$ is the target, such that $\alpha_{t,t} = 1$ and $0 \leq \alpha_{t,r} \leq 1$.

The DMF framework can be used with models with different prediction functions $\hat{y}_{t,r}$. To illustrate this, let us have a look into how a model like the one from Equation 5 can be learned using this framework. DMF associates one latent feature vector $\varphi_r(x)$ with each instance $x$ for each relation $r = 1, \ldots, R$. Accordingly, different feature matrices $\Phi_{t,r}$ are associated with each relation $r = 1, \ldots, R$, one per target $t = 1, \ldots, R$. The prediction function in Equation 5 can be rewritten under the DMF framework as in Equation 7.

$$\hat{y}_{t,r}(x_1, x_2) := \varphi_t(x_1)^\top \Phi_{t,r} \varphi_t(x_2) \quad (7)$$

The DMF loss decomposes over $t$ and each component can be optimized independently of each other; this is equivalent to $R$ independent models, one for each target relation. Another point worth noting is that the $\alpha_{t,r}$ relation weights are crucial for this model, e.g., setting all of them to 1 is the same as learning the same model $R$ times (up to a random initialization).

To make this argument more clear, let us revisit the social media example. In that example there are two entity types, namely users $U$ and news items $N$ and three relations: follows $F := U \times U$, the social relationship $S := U \times U$ and the product consumption (reading of news items) $C := U \times N$. A state-of-the-art multi-factorization model like, for instance RESCAL, would define latent features for users $\varphi(U)$, news

[1]The authors refer to the model as a multiple order factorization [9].

items $\varphi(N)$ as well as for the relations $\Phi_F$, $\Phi_S$ and $\Phi_C$ and learn them as in Equation 8 (regularization terms are omitted here to avoid clutter).

$$(\varphi^*(U), \varphi^*(N), \Phi_F^*, \Phi_S^*, \Phi_C^*) := \underset{\varphi(U),\varphi(N),\Phi_F,\Phi_S,\Phi_C}{\arg\min} L_F(D_F, \hat{y}_F(\cdot; \varphi(U), \Phi_F)) + L_S(D_S, \hat{y}_S(\cdot; \varphi(U), \Phi_S)) + L_C(D_C, \hat{y}_C(\cdot; \varphi(U), \varphi(N), \Phi_C)) \quad (8)$$

This way, the same user features $\varphi^*(U)$ are used for making predictions for all relations and thus we will refer to this strategy as *complete sharing*. Now, suppose one uses different latent features for different target relations and $\varphi_F(U)$, $\varphi_S(U)$, $\varphi_C(U)$ denote the user features used for making predictions for relations $F$, $S$ and $C$ respectively. Then, it would be possible to learn features such that

$$\varphi_F^*(U) := \underset{\varphi(U)}{\arg\min} L_F(D_F, \hat{y}_F(\cdot; \varphi(U), \Phi_F))$$
$$\varphi_S^*(U) := \underset{\varphi(U)}{\arg\min} L_S(D_S, \hat{y}_S(\cdot; \varphi(U), \Phi_S))$$
$$\varphi_C^*(U) := \underset{\varphi(U)}{\arg\min} L_C(D_C, \hat{y}_C(\cdot; \varphi(U), \varphi(N), \Phi_C))$$

while models that follow the complete sharing strategy and learn parameters like in Equation 8 are constrained to solutions of the form

$$\varphi_F(U) = \varphi_S(U) = \varphi_C(U) = \varphi^*(U).$$

However, when learning the parameters for a given relation, it is important to exploit the information about the other relations. Thus, we can reformulate the multi-target factorization problem as a set of single target problems, one for each target relation. This way, the parameters for relation $F$ acting as a target relation are learned as:

$$(\varphi_F^*(U), \Phi_F^*) := \underset{\varphi_F(U),\Phi_F}{\arg\min} L_F(D_F, \hat{y}_{F,F}(\cdot; \varphi_F(U), \Phi_{F,F})) + \alpha_{F,S} L_S(D_S, \hat{y}_{F,S}(\cdot; \varphi_F(U), \Phi_{F,S})) + \alpha_{F,C} L_C(D_C, \hat{y}_{F,C}(\cdot; \varphi_F(U), \varphi_F(N), \Phi_{F,C})) .$$

The same way, when relation $S$ is the target, the model looks like

$$(\varphi_S^*(U), \Phi_S^*) := \underset{\varphi_S(U),\Phi_S}{\arg\min} L_S(D_S, \hat{y}_{S,S}(\cdot; \varphi_S(U), \Phi_{S,S})) + \alpha_{S,F} L_F(D_F, \hat{y}_{S,F}(\cdot; \varphi_S(U), \Phi_{S,F})) + \alpha_{S,C} L_C(D_C, \hat{y}_{S,C}(\cdot; \varphi_S(U), \varphi_S(N), \Phi_{S,C})) .$$

Analogously, the same is done for relation $C$. Since there are three relations, each user $u \in U$ and news item $n \in N$ is associated with three latent feature vectors, each corresponding to the case where each relation acts as target. This can be seen in Figure 2. There, one can see that when one relation acts as a target the other ones are useful for regularizing the parameters for predicting it.

Since the *follows* ($F$) relation is a relation between users and users, one does not need the feature vectors of news items $\varphi_F(n)$ for *predicting* it. However, these parameters are useful when *learning* user features $\varphi_F(u)$ since news item

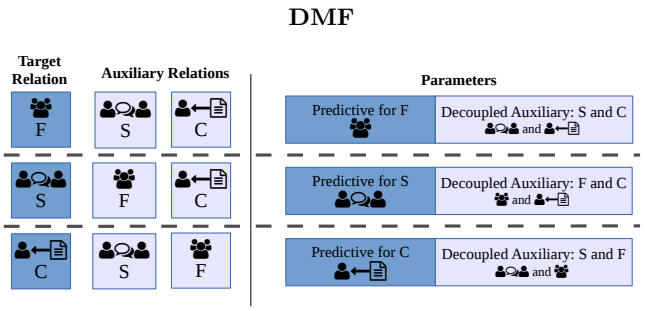


Figure 2: **DMF parameters for the social media example.**

features are needed to regularize user features using the *consumes* ($C$) relation. Hence we dub such parameters *auxiliary parameters*. In Figure 2, predictive parameters are depicted in a darker blue color and the auxiliary ones in a lighter gray.

## CATSMF

One issue with DMF is that the number of parameters to be learned grows by a factor $R$ of the number of relations in the dataset. When relation feature vectors are used, DMF has in total $R^2k + R|\mathcal{E}|k$ parameters. This is of course undesirable from the scalability point of view. Furthermore, the fact that individual models are completely decoupled from each other prevents that one benefits from the learning process of the other. To tackle both issues, we propose to couple the models by sharing the parameters used for auxiliary relations. We call this approach the *Coupled Auxiliary and Target Specific Features Multi-target Factorization (CATSMF)* and it represents our core contribution. The prediction model from Equation 5 written using the CATSMF approach is as follows:

$$\hat{y}_{t,r}(x_1, x_2) := \varphi_{t \cdot \delta(x_1 \in \mathcal{E}_t^{(1)})}(x_1)^\top \Phi_{r,\delta(t=r)} \varphi_{t \cdot \delta(x_2 \in \mathcal{E}_t^{(2)})}(x_2) \quad (9)$$

where $\mathcal{E}_r^{(1)} \subseteq \mathcal{E}$ and $\mathcal{E}_r^{(2)} \subseteq \mathcal{E}$ are the sets of entities that could possibly occur as the subjects and the objects of relation $r$, respectively, as defined in Section 2. This means that entities occurring within the target relation $t$ are associated with target specific features $\varphi_t$, while entities that do not occur within the target relation $t$ are associated with auxiliary features $\varphi_0$ (pooled over all target relations). Every relation $r$ has two feature matrices: one when used as target $\Phi_{r,1}$ and another one when used as auxiliary relation $\Phi_{r,0}$.

While DMF defines a full set of parameters for each relation, CATSMF defines parameters needed to make the predictions for each target relation, plus one full set of auxiliary ones. For example, if a given entity $x$ does not occur in a relation $t$, i.e., $x \notin \mathcal{E}_t^{(1)} \cup \mathcal{E}_t^{(2)}$, then $\hat{y}_{t,t}$ is never computed for $x$ and thus $\varphi_t(x)$ is never used and can be dropped. For instance, if $r$ is the relation *father-of*, $\mathcal{E}_r^{(1)}$ and $\mathcal{E}_r^{(2)}$ both correspond to the subset of persons, but not, say, locations, and if $r'$ is the relation *capital-of*, $\mathcal{E}_{r'}^{(1)}$ corresponds to the subset of cities while $\mathcal{E}_{r'}^{(2)}$ corresponds to the subset of countries, but not persons. This means, that for two given entities, a person *John* and location *Berlin*, $\varphi_{\text{capital-of}}(\text{John})$ and $\varphi_{\text{father-of}}(\text{Berlin})$ need not be computed. Besides leading to a lower number of parameters, taking into consideration entity

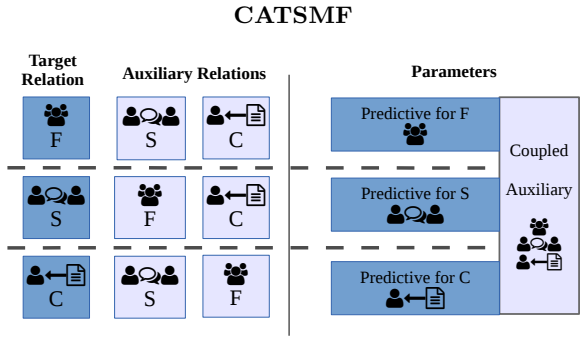


Figure 3: **CATSMF parameters for the social media example. Please note that the auxiliary parameters are shared across the target relations.**

types can lead to better predictive performance as observed in Section 5 among the results of our experiments.

Figure 3 shows the CATSMF setup for the social media example. Note how the number of parameters is reduced in comparison to DMF by sharing auxiliary parameters. In Figure 2 one can see that there is a lot of redundancy in DMF regarding auxiliary parameters. There are two copies of auxiliary parameters for item features and two copies of each relation auxiliary features. What CATSMF does is essentially to define one set of auxiliary parameters and share them through the cases of different target relations.

CATSMF has two main advantages over DMF: (i) since the auxiliary parameters are shared across the models for different target relations, such models are coupled and can profit from each other. (ii) CATSMF allows for a lower number of parameters. The number of latent features needed by CATSMF is $2Rk + \sum_{r=1}^{R} |\mathcal{E}_r|k$, where we define $\mathcal{E}_r := \mathcal{E}_r^{(1)} \cup \mathcal{E}_r^{(2)}$ to simplify the notation. The lower $\sum_{r=1}^{R} |\mathcal{E}_r|$, the bigger the savings in the number of parameters. Even in the worst case scenario, where there is no entity type information available, i.e., if $\mathcal{E}_r = \mathcal{E}$ for all $r = 1, \ldots, R$, the number of parameters required by CATSMF is $2Rk + R|\mathcal{E}|k$. This means that, while the relationship between the number of parameters and the amount of relations is quadratic for DMF, for CATSMF it is linear.

CATSMF is learned through stochastic gradient descent as shown in Algorithm 1. The algorithm starts by initializing the parameters, drawing them from a 0-mean normal distribution (lines 2–7). Then, a target relation $t$ is uniformly sampled and a stochastic gradient descent update is made in one observation of $t$ (lines 9–10) according to Algorithm 2. Finally, another relation $r$ is uniformly sampled and an update on this relation, acting as an auxiliary relation for $t$, is performed (lines 11–12). We do this oversampling of target specific parameters to guarantee that they are more often updated than the auxiliary ones, which leads to faster empirical convergence.

The parameter update is described in Algorithm 2. The first step is to uniformly sample an observation $(x_1, x_2, y) \in D_r$ (line 2). The next step is to determine the parameters to estimate $\hat{y}_{t,r}(x_1, x_2)$ that will be updated (lines 3, 5 and 7). If $r = t$, then it plays a target relation role and only the target specific parameters regarding $t$, namely $\Phi_{t,1}$, $\varphi_t(x_1)$ and $\varphi_t(x_2)$ are updated. In case $r \neq t$, then $r$ plays the role of an auxiliary relation. In this case the auxiliary features $\Phi_{r,0}$ are used. If $x_1$ is among the entities related by $t$, i.e., $x_1 \in \mathcal{E}_t$, then $\varphi_t(x_1)$ is used, otherwise the auxiliary features $\varphi_0(x_1)$ are in place. We proceed analogously for $x_2$. Finally, the chosen parameters are updated with a stochastic gradient descent step (lines 4, 6 and 8).

**Algorithm 1 CATSMF**

1: **procedure** LEARNMULTITARGET
**input:** number of relations $R$, training data $\{D_r\}_{r=1,\ldots,R}$, set of entities $\mathcal{E}$, set of entities that possibly could occur in relation $r$: $\{\mathcal{E}_r\}_{r=1,\ldots,R}$, learning rate $\eta$, and regularization constants $\lambda$

2: $\forall_{x \in \mathcal{E}} \;\; \varphi_0(x) \sim \mathcal{N}(0, \sigma^2)$
3: **for** $r = 1, \ldots, R$ **do**
4: $\forall_{x \in \mathcal{E}_r} \;\; \varphi_r(x) \sim \mathcal{N}(0, \sigma^2)$
5: $\Phi_{r,0}(x) \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
6: $\Phi_{r,1}(x) \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$
7: **end for**
8: **repeat**
9: $t \sim \text{Uniform}(1, R)$
10: $(\varphi, \Phi) = \text{UpdateModel}(t, t, D_t, \varphi, \Phi, \eta, \lambda_t)$
11: $r \sim \text{Uniform}(1, R)$
12: $(\varphi, \Phi) = \text{UpdateModel}(t, r, D_r, \varphi, \Phi, \eta, \lambda_t)$
13: **until** convergence
14: **end procedure**

**Algorithm 2 CATSMF Stochastic Gradient Descent Update**

1: **procedure** UPDATEMODEL
**input:** target relation $t$, auxiliary relation $r$, observations about relation $r$: $D_r$, set of entity features $\varphi$, set of relation features $\Phi$, learning rate $\eta$, and regularization constant $\lambda_t$
**output:** updated entity features $\varphi$ and updated relation features $\Phi$

2: $(x_1, x_2, y) \sim \text{Uniform}(D_r)$
3: $r' \leftarrow t\delta(x_1 \in \mathcal{E}_t)$
4: $\varphi_{r'}(x_1) \leftarrow \varphi_{r'}(x_1) - \eta \left( \frac{\partial \ell_r(y, \hat{y}_{t,r}(x_1, x_2))}{\partial \varphi_{r'}(x_1)} + \lambda_t \varphi_{r'}(x_1) \right)$
5: $r' \leftarrow t\delta(x_2 \in \mathcal{E}_t)$
6: $\varphi_{r'}(x_2) \leftarrow \varphi_{r'}(x_2) - \eta \left( \frac{\partial \ell_r(y, \hat{y}_{t,r}(x_1, x_2))}{\partial \varphi_{r'}(x_2)} + \lambda_t \varphi_{r'}(x_2) \right)$
7: $r' \leftarrow r\delta(t = r)$
8: $\Phi_{r'} \leftarrow \Phi_{r'} - \eta \left( \frac{\partial \ell_r(y, \hat{y}_{t,r}(x_1, x_2))}{\partial \Phi_{r'}} + \lambda_t \Phi_{r'} \right)$
9: **return** $(\varphi, \Phi)$
10: **end procedure**

### *Setting up CATSMF*

Often overlooked in the multi-relational factorization literature are bias terms. We use target-specific and auxiliary bias terms. The prediction function is the following:

$$\hat{y}_{t,r}(x_1, x_2) := b_{r,\delta(t=r)} + b_{t \cdot \delta(x_1 \in \mathcal{E}_t)}(x_1) + b_{t \cdot \delta(x_2 \in \mathcal{E}_t)}(x_2) + \varphi_{t \cdot \delta(x_1 \in \mathcal{E}_t)}(x_1) \Phi_{r,\delta(t=r)} \varphi_{t \cdot \delta(x_1 \in \mathcal{E}_t)}(x_2) \tag{10}$$

where $b_{r,\delta(t=r)}, b_{t\delta(x_1 \in \mathcal{E}_t)}(x_1), b_{t\delta(x_2 \in \mathcal{E}_t)}(x_2)$ are bias terms. The parameters in this prediction function should be opti-

mized for the task at hand, which is to make predictions based on positive only observations. In [6], it is empirically shown that the BPR optimization criterion (BPR-Opt) [16] is suitable for this task.

Let $\sigma(x) = \frac{1}{1+e^{-x}}$ be the sigmoid function, BPR-Opt is an instance of a pairwise loss and can be defined for a general multi-relational learning task as follows:

$$\text{BPR-Opt}_r(D_r, \hat{y}_{t,r}) := \sum_{(x_1,x_2,1)\in D_r^{\text{train}}} \sum_{(x_1,x_2',1)\in\mathcal{X}_r\times\{1\}\setminus D_r^{\text{train}}} \ln\sigma(\hat{y}_{t,r}(x_1,x_2) - \hat{y}_{t,r}(x_1,x_2')).$$

# 5. EXPERIMENTAL EVALUATION

In this section, CATSMF and DMF are compared against each other and against state-of-the-art competitors. More specifically, we examine the impact of using target specific parameters as well as of considering target and auxiliary roles for relations.

The evaluation assesses the behavior of CATSMF on practical Web applications using three large Web datasets. The datasets used in the experiments are described next, followed by the metrics and evaluation protocol and the state-of-the-art baselines used in the experiments. We conclude the section presenting the detail of the results of our empirical study.

## Datasets

In our experiments, we used three large Web datasets collected from DBpedia, Wikipedia and BlogCatalog. DBpedia is one of the central interlinking-hubs of the emerging Web of Data,[2] which makes it really attractive to evaluate multi-relational learning approaches. The Wikipedia-SVO dataset has one of the highest number of relations among published multi-relational datasets [9]. The BlogCatalog dataset [21] has been used in the literature to evaluate recommender systems that exploit social network information [10, 21]. The three Web datasets are detailed as follows:

- **DBpedia** dataset corresponds to a sample of 625,680 triples from the *DBpedia Properties* in English[3]. It consists of 269,862 entities and 5 relations regarding the music domain. Such relations are: `artist`, `genre`, `composer`, `associated_band`, and `associated_musical_artist`.
- **Wikipedia-SVO** [9] depicts word relationships in the form of subject-verb-object triples extracted from over two million Wikipedia articles, where the verbs play the role of the relationship. It consists of 1,300,000 triples about 4,547 relationships and 30,605 entities.
- **BlogCatalog**[4] [21] is a large blogging website with social network features. The dataset consists of two relations, with one relation between users and blogs indicating which blogs the users find interested and the social relation between users and other users. The task at hand is to recommend both interesting blogs and potential new friends to users. Note that previous work on this dataset [10, 21] focused on the single target task of using the social information to recommend blogs. There is a total of 10,312 users and 39 blogs.

[2] `http://lod-cloud.net`

[3] `http://downloads.dbpedia.org/3.6/`

[4] `http://www.blogcatalog.com/`

## Evaluation Protocol and Metrics

The dataset is split into training, validation, and test set. First, 10% of the positive tuples are randomly selected and assigned to the test set. Then, we randomly sample 10% of the remaining ones to form the validation set. The remaining triples are used for training. To reduce variability, 10-fold cross-validation was performed. The results reported are the average over the rounds considering 99% confidence intervals. For this evaluation, we follow a protocol based on [4] as described next.

For each relation $r$ and entity $x$ on the test set:

1. First, we sample from $r_x^- \subseteq \{(x,x_2,y)|(x,x_2,y) \notin D_r^{\text{train}} \cup D_r^{\text{test}}\}$, i.e. the set of unobserved triples in the knowledge base.
2. Then, we compute the score for the $|r_x^-|$ negative triples and for each of the positive ones in the test set: $r_x^+ = \{(x,x_2,y)|(x,x_2,y) \in D_r^{\text{test}}\}$.
3. Finally, we measure the **precision** and **recall** at $n = 1,\ldots,10$ on this list of triples and report the results by plotting the corresponding precision-recall curves [2].

*Parameter Setting.* For each dataset, split and model, we tune the hyperparameters using the train and validation set through grid-search. Next, the models were retrained on both train and validation sets and evaluated on test partition. The results reported correspond to the performance of the methods on the test set only. This process was performed for *all* the models in the evaluation, including the baselines. Regarding hyperparameter values, the number of latent features $k$ was searched in the range $\{10, 25, 50\}$ for all baselines and variants of our approach. The values for $\alpha_{t,r}$, $\lambda_r$ and $\eta$ were searched in $\{0.25, 0.5, 0.75\}$, $\{0.0001, 0.001, 0.01\}$ and $\{0.0005, 0.005, 0.05\}$, respectively. All the hyperparameters for the baselines were searched in the ranges suggested by their respective authors in their papers.

## Comparison against the state-of-the-art

As far as the approach proposed here is concerned we want to make sure that any effects observed come from the usage of predictive and auxiliary features and not from a specific loss or how relation feature matrices look like. Thus, three variants of the same prediction model are evaluated. They are detailed as follows:

- **Shared-Diag-BPR** uses the *complete sharing* strategy. This is how state-of-the-art methods approach model parametrization. **Shared-Diag-BPR** can be seen as **RESCAL** with a diagonal matrix for relation features and optimized for BPR-Opt instead of the L2 loss.
- **DMF-Diag-BPR** comprises a set of decoupled models (i.e., no parameter sharing between them), one for each target relation as in Equation 7.
- **CATSMF-Diag-BPR** is the core contribution of this paper, that uses the parametrization from Equation 10, with target-specific parameters and shared ones for auxiliary relations.

| Method | Prediction function $\hat{y}_r(x_1, x_2)$ | Relation loss | Relation Features | Target Parameters |
|---|---|---|---|---|
| **Shared-Diag-BPR** | $b_r + b(x_1) + b(x_2) + \varphi(x_1)^\top \Phi_r \varphi(x_2)$ | BPR | Diagonal Matrix | Complete Sharing |
| **DMF-Diag-BPR** | $b_{r,1} + b_r(x_1) + b_r(x_2) + \varphi_t(x_1)^\top \Phi_r \varphi_t(x_2)$ | BPR | Diagonal Matrix | DMF |
| **CATSMF-Diag-BPR** | $b_{r,1} + b_r(x_1) + b_r(x_2) + \varphi_{t\delta(x_1 \in \mathcal{E}_t)}(x_1)^\top \Phi_{r,\delta(t=r)} \varphi_{t\delta(x_2 \in \mathcal{E}_t)}(x_2)$ | BPR | Diagonal Matrix | CATSMF |
| **RESCAL** | $\varphi(x_1)^\top \Phi_r \varphi(x_2)$ | L2 | Full Matrix | Complete Sharing |
| **MOF-SRP** | $\mathbf{a}\Phi_r\mathbf{a}^\top + \varphi(x_1)\Phi_r\mathbf{b}^\top + \mathbf{b}\Phi_r\varphi(x_2)^\top + \varphi(x_1)^\top\Phi_r\varphi(x_2)$ | Logistic | Outer Product of Latent Vectors | Complete Sharing |

Table 1: **Models used in the CATSMF and DMF evaluation.**

Our approaches are also compared against the following state-of-the-art models:

- **RESCAL** [13] uses the *complete sharing* strategy and can be described as *Shared-Full-L2*. This model does not make use of specific target features and uses full matrices for relation features. Specific relations are optimized for the L2 loss;
- **MOF-SRP** [9] also follows the *complete sharing* strategy and uses full matrices for relation features but represents them by outer products of one dimensional arrays, in order to require fewer parameters. The model is optimized for the logistic loss.

Table 1 presents a summary of the approaches evaluated.

## Results and Discussion

In the DBpedia and BlogCatalog datasets we set $\alpha_{t,t} = 1$ and estimated both the $\alpha_{t,r}$ and the $\lambda_t$ values through grid search. On the Wikipedia-SVO dataset is infeasible to estimate each $\alpha_{t,r}$ value through grid search, given the number of relations in this dataset. Therefore, we set each $\alpha_{t,r} = a$, for $t \neq r$, where $a$ is a hyperparameter estimated on validation data using grid search. We did the same for the regularization constants, i.e., setting all $\lambda_t = \lambda$ and optimizing $\lambda$. Figure 4 shows the precision-recall curves with the results of our evaluation.

When analyzing the results we observe that our approach excels in the three Web datasets used in the empirical evaluation as seen in Figure 4. However, since the models evaluated use different parametrization, prediction, and loss functions, we need to explore in more detail which aspects of the models are responsible for the relative differences in performance. Therefore, to answer the question: *what is the impact on prediction performance of using target specific parameters, while using the same prediction function and the same relation specific losses?* the first important aspect to observe is how the CATSMF-Diag and DMF-Diag approaches compare to the Shared-Diag approach. This is because they are essentially the same model where each individual relation is optimized for the same loss function and the only differences between them are whether they use target specific predictive parameters or not, and the corresponding strategy used to this end. When comparing those three approaches, one can clearly see that both DMF and CATSMF outperform the complete sharing approach.

The results show that using target specific parameters improves over the complete parameter sharing scenario while using both shared and target specific parameters gives an even stronger performance boost.

In the comparison against the state-of-the-art approaches, the differences in performance can be mostly explained by the usage of different loss functions for individual relations. The BPR loss deals better with the scenario where only positive observations are available. On the other hand, the pairwise interactions modeled by MOF-SRP seem to play an important role in the DBPedia dataset, which explains the good performance of this model there. The fact that modeling pairwise interactions leads to better predictive performance on RDF datasets has been observed before in [6]. It is important to note that RESCAL and MOF-SRP also can be used within the target specific parameter framework offered by CATSMF and DMF.

CATSMF has clearly the best performance in the BlogCatalog and Wikipedia-SVO datasets.[5]

We believe that the poor performance of RESCAL on the BlogCatalog dataset is due to the optimization for the squared error since it has been observed that models optimized for the BPR loss perform much better on this particular dataset [10].

We observe that MOF-SRP is not as competitive on the BlogCatalog dataset as it is on the other ones. One possible explanation is that this model does not take into account entity type information. This means that when learning on the User-Blog relation, the MOF-SRP assumes that users are also potential items to be recommended. As reported by [9], negative examples are sampled when learning the model, but since it does not differentiate between *users* and *blogs* and there are 10,312 users and only 39 blogs, one can expect that approximately 99.6% of the sampled negative examples are trivial ones containing recommendations of users. One can see this when looking into the performance on the individual relations. On the social User-User relation, MOF-SRP achieves 0.901 AUC[6] against 0.961 AUC of CATSMF. On the User-Blog relation however the AUC for MOF-SRP

[5]In addition to the results reported here, we reproduced the same experiment on the Wikipedia-SVO dataset performed by [9], where the hit rate at the top 5% (referred in their paper as p@5) and 20% (referred to in the original paper as p@20) is measured. CATSMF achieves a p@5 of 0.74 and a p@20 of 0.95, while MOF-SRP is reported to achieve 0.75 and 0.95, respectively.

[6]AUC: area under the ROC curve. [2]

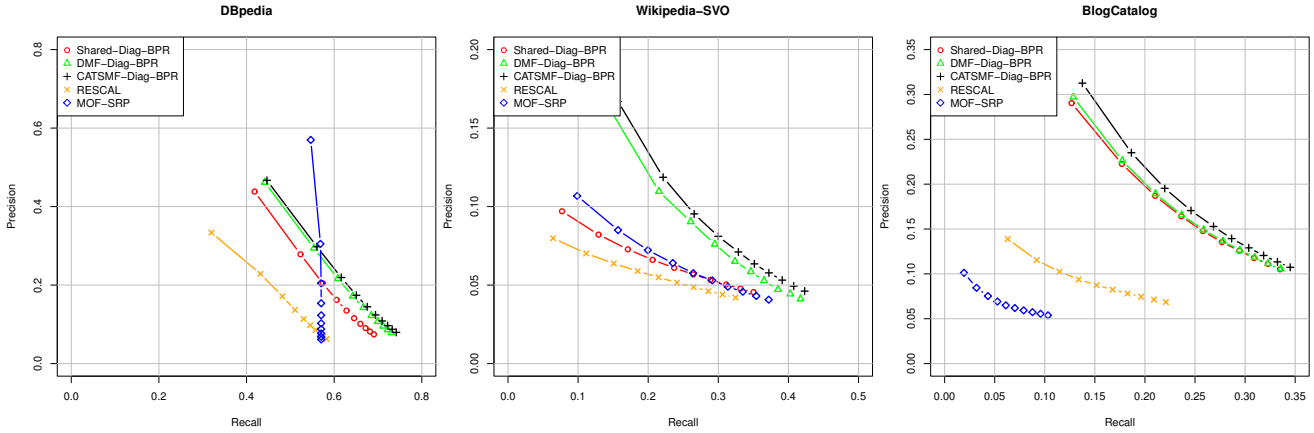


Figure 4: **Performance of CATSMF against state-of-the-art baselines on the three Web datasets.**

is 0.481 against 0.825 of CATSMF. These results suggest MOF-SRP was not able to learn accurately the information about entity types for this dataset.

### *Impact of the relation weights.*

We discuss here the impact of (i) using auxiliary relations with CATSMF and (ii) optimizing each $\alpha_{t,r}$ value individually instead of setting $\alpha_{t,r} = a$ and optimizing $a$ (i.e., use the same value for all parameters). In Figure 5, we see the performance of CATSMF-Diag-BPR model on the BlogCatalog dataset under these various settings. This dataset has two relations, the social relationship $s$ and the interest of users in blogs $i$, thus leaving us with just two $\alpha$ values to optimize: $\alpha_{s,i}$ and $\alpha_{i,s}$, since $\alpha_{i,i} = \alpha_{s,s} = 1$. Each curve denotes a different setting of each $\alpha_{t,r}$, i.e., setting all of them to the same value or optimizing them individually. One can see that by using no auxiliary relations ($\alpha = 0$) the model presents its worst performance whereas by optimizing each relation weight individually, it exhibits its best performance. Figure 5 also provides empirical evidence that, in datasets with a large number of relations, where setting each relation weight individually might be costly or even infeasible, setting them to the same value might be a good compromise for the tradeoff between predictive performance and cost of model selection.

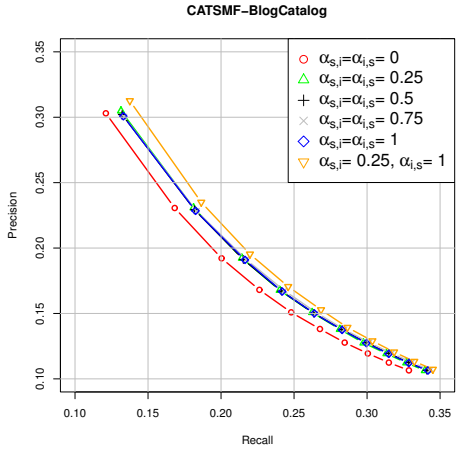


Figure 5: **Performance of CATSMF on the BlogCatalog dataset for different values of $\alpha$.**

### *Runtime.*

Here we report the average runtime over 10 single thread runs on the DBpedia dataset on a Xeon E5620 2.40GHz CPU. The average duration is 754 seconds for CATSMF-Diag-BPR, 7039.5 seconds for RESCAL and 77406.25 seconds or, approximately, 21 hours for MOF-SRP on the DBpedia dataset, which shows that CATSMF-Diag-BPR scales much better w.r.t. runtime while providing very competitive prediction performance. There are two main reasons that can explain the better runtime performance of CATSMF-Diag-BPR: (i) the fact that CATSMF-Diag-BPR uses a diagonal matrix for relation features whereas RESCAL uses a full matrix and MOF-SRP a matrix represented as outer products of feature vectors and (ii) we learn CATSMF-Diag-BPR using the scalable stochastic gradient descent learning algorithm.

As discussed in Section 3, using a diagonal matrix as relation features may not be the best choice in terms of prediction performance. However, as shown in Figure 4 the target-specific strategy of CATSMF-Diag-BPR improves the results making it competitive against state-of-the-art models while still having much lower runtime.

### *Reproducibility of the experiments.*

DMF and CATSMF implementations are available online.[7] For RESCAL and MOF-SRP we used the implementations provided by the authors.

## 6. CONCLUSION AND FUTURE WORK

In this work we argue and show empirically how multi-relational factorization models can benefit from using different parametrizations of prediction functions for individual target relations. We first introduce a naive set of decoupled models, one for each target relation called DMF, followed by a more memory-efficient variant with shared auxiliary parameters: CATSMF, which has fewer parameters thus scales better. The novelty of DMF and CATSMF lies in the fact that they learn different sets of parameters for reconstructing particular target relations. In contrast to the trivial DMF solution of learning one model per target, where all models are completely decoupled from each other, CATSMF defines parameters to be used when a given relation plays an

[7]Code available at `http://ismll.de/catsmf`

auxiliary role, which are shared among all different models for the various targets.

Our experiments show that (i) CATSMF is able to scale to large datasets better than state-of-the-art models, while still achieving competitive predictive performance; (ii) CATSMF and DMF are always at least as good as the standard approach of using the same set of parameters for all target relations, but often outperforms it; (iii) CATSMF outperforms competitor models in the Linked Open Data mining, natural language processing, and recommender systems tasks.

We are currently exploring how to effectively estimate the relation weights $\alpha_{t,r}$ and the regularization constants $\lambda_t$ from the data, considering that setting them through model selection might be infeasible even for a moderate number of relations. Two promising approaches are (i) based on adaptive regularization [15] and (ii) learning the model in a Bayesian framework and estimate the hyperparameters using hierarchichal models similar to what Singh and Gordon propose [20]. As future work, we plan to investigate a more memory efficient CATSMF variant, e.g., by reducing the number of parameters to be learned. One possible alternative to this end would be to introduce an $\ell_1$ regularizer so as to remove some of the small parameters and trim the model even further. An additional direction for future work is the extension of our framework for streaming data scenarios, e.g., [5], where the model parameters have to be learned online without compromising ranking performance.

**Acknowledgments:** The authors would like to thank Nicolas Schilling, Francesco Calabrese, Charles Jochim, Dimitrios Mavroeidis, and Fabio Pinelli for insightful comments on preliminary versions of the paper. This work was funded in part by the L3S IAI-Reasearch Grant: *FizzStream* and by Deutsche Forschungsgemeinschaft within the project Multi-relational Factorization Models (`www.ismll.de/projekte/dfg_multirel_en.html`).

# 7. REFERENCES

[1] E. Acar, T. G. Kolda, and D. M. Dunlavy. All-at-once Optimization for Coupled Matrix and Tensor Factorizations. In *MLG'11: Proceedings of Mining and Learning with Graphs*, August 2011.

[2] R. A. Baeza-Yates and B. A. Ribeiro-Neto. *Modern Information Retrieval - the concepts and technology behind search, Second edition*. Pearson Education Ltd., Harlow, England, 2011.

[3] A. Bordes, X. Glorot, J. Weston, and Y. Bengio. A semantic matching energy function for learning with multi-relational data. *Machine Learning*, 94(2), 2014.

[4] P. Cremonesi, Y. Koren, and R. Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *Proceedings of the fourth ACM conference on Recommender systems*, RecSys '10, 2010.

[5] E. Diaz-Aviles, L. Drumond, L. Schmidt-Thieme, and W. Nejdl. Real-time Top-n Recommendation in Social Streams. In *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, 2012.

[6] L. Drumond, S. Rendle, and L. Schmidt-Thieme. Predicting RDF triples in incomplete knowledge bases with tensor factorization. In *Proceedings of the 27th Annual ACM Symposium on Applied Computing*, SAC '12, 2012.

[7] L. Getoor and B. Taskar, editors. *Introduction to Statistical Relational Learning*. The MIT Press, 2007.

[8] R. Harshman. Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multi-mode factor. In *UCLA Working Papers in Phonetics*, 1970.

[9] R. Jenatton, N. L. Roux, A. Bordes, and G. Obozinski. A latent factor model for highly multi-relational data. In *Advances in Neural Information Processing Systems (NIPS)*. 2012.

[10] A. Krohn-Grimberghe, L. Drumond, C. Freudenthaler, and L. Schmidt-Thieme. Multi-relational matrix factorization using bayesian personalized ranking for social network data. In *Proceedings of the fifth ACM International Conference on Web Search and Data Mining WSDM '12*, 2012.

[11] Y.-R. Lin, J. Sun, P. Castro, R. Konuru, H. Sundaram, and A. Kelliher. MetaFac: Community Discovery via Relational Hypergraph Factorization. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '09, 2009.

[12] H. Ma, H. Yang, M. R. Lyu, and I. King. SoRec: social recommendation using probabilistic matrix factorization. In *Proceeding of the 17th ACM conference on Information and knowledge management*, CIKM '08, 2008.

[13] M. Nickel, V. Tresp, and H. Kriegel. A Three-Way Model for Collective Learning on Multi-Relational Data. In *Proceedings of the 2011 International Conference on Machine Learning (ICML)*, 2011.

[14] M. Nickel, V. Tresp, and H.-P. Kriegel. Factorizing YAGO: scalable machine learning for linked data. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, 2012.

[15] S. Rendle. Learning recommender systems with adaptive regularization. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, 2012.

[16] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme. BPR: Bayesian personalized ranking from implicit feedback. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, UAI '09, 2009.

[17] M. Richardson and P. Domingos. Markov logic networks. *Machine learning*, 62(1), 2006.

[18] W. Shen, J. Wang, P. Luo, and M. Wang. LINDEN: linking named entities with knowledge base via semantic knowledge. In *Proceedings of the 21st international conference on World Wide Web*, WWW '12, 2012.

[19] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proceeding of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2008.

[20] A. P. Singh and G. J. Gordon. A Bayesian matrix factorization model for relational data. In *Proceedings of the Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.

[21] L. Tang and H. Liu. Relational learning via latent social dimensions. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining KDD '09*, 2009.

[22] Y. Zhang, B. Cao, and D.-Y. Yeung. Multi-Domain Collaborative Filtering. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI)*, 2010.